CS 310 | Spring 2022 | Unit 5 | Survey of Logical Languages

Topics: *Logical programming, language translation, predicate logic*

Follow the instructions using source code. Submit the indicated files (📎) via the assignments in the eCampus lecture section by the due dates on the course calendar. For each submitted file, include your full name as author and a statement acknowledging that your work complies with the academic integrity policy.

If you are pair programming, you and your peer each submit the indicated files independently and are graded independently. For each submitted file, also include your peer's full name as co-author and a description of your and your peer's contributions.

INSTRUCTIONS

A Choose **one** of the two predicates below and follow the steps for your chosen predicate.
   1 If you are pair programming, you and your peer can choose the same or different predicates.
   2 For **15% bonus** on the unit, follow the steps for **both** predicates, not just for one chosen predicate. The better of the two implementations determines your unit grade, while the worse of the two determines your bonus grade, proportionally.

DECK PIVOTING PREDICATE

B Write a 📎 script in **Prolog** (SWI dialect) with all the facts and rules necessary to implement the `pivot/2` predicate (**50%**).
   1 If you are pair programming, you can complete this implementation with a peer or individually.
   2 Assume `Before` and `After` are simple lists of atoms and at least one of them is instantiated when the predicate is queried.
   3 The goal `pivot(Before, After)` is true only when `Before` and `After` have the same elements but the first and second halves of equal size are transposed. With an odd number of elements, the middle element is fixed between the halves.
   4 These examples cover all 4 instantiation cases: 3 are valid (✓) and must be supported, but 1 is undefined (?).
      i ✓ Both variables are instantiated: `pivot([1,2,3,4,5,6,7], [5,6,7,4,1,2,3]).` yields a `true`/`yes` result.
      ii ✓ Only `Before` is instantiated: `pivot([1,2,3,4,5,6], R).` unifies `R = [4,5,6,1,2,3]` as its one result.
      iii ✓ Only `After` is instantiated: `pivot(L, [1,2]).` unifies `L = [2,1]` as its one result.
      iv ? Neither variable is instantiated: `pivot(L, R).` is undefined (since results are generated arbitrarily).

C Write a 📎 class in **Java** equivalent in functionality to the Prolog script defined in step B (**50%**).
   1 You must complete this implementation individually, even if you are pair programming.
   2 All valid instantiation cases must be supported using one or more overloaded methods.
   3 The formal parameters of a method correspond to the instantiated variables of that case.
   4 The return value of a method corresponds to the result of that case.
      i For cases which only yield `true`/`yes` or `false`/`no` results, define a method which returns a boolean.
      ii For cases which unify a variable, define a method which returns the unified result of that variable.

ITERATED LOGARITHM PREDICATE

D Write a 📎 script in **Prolog** (SWI dialect) with all the facts and rules necessary to implement the `lgstar/2` predicate (**50%**).
   1 If you are pair programming, you can complete this implementation with a peer or individually.
   2 Assume `N` is a number, `Iterations` is an integer, and at least `N` is instantiated when the predicate is queried.
   3 The goal `lgstar(N, Iterations)` is true only when `Iterations` equals the base-2 iterated logarithm $\lg^*(N)$, which is different from the base-10 iterated logarithm $\log^*(N)$ and requires a change of base accordingly.
   4 These examples cover all 4 instantiation cases: 2 are valid (✓) and must be supported, but 2 are undefined (?).
      i ✓ Both variables are instantiated: when intermediate numbers are floored to integers `lgstar(70000, 4).` yields a `true`/`yes` result, otherwise `lgstar(70000, 5).` yields a `true`/`yes` result. Either interpretation is acceptable.
      ii ✓ Only `N` is instantiated: per interpretation `lgstar(70000, A).` unifies either `A = 4` or `A = 5` as its one result.
      iii ? Only `Iterations` is instantiated: `lgstar(X, 3).` is undefined (since there are infinite unifications).
      iv ? Neither variable is instantiated: `lgstar(X, A).` is undefined (since results are generated arbitrarily).

E Write a 📎 class in **Java** equivalent in functionality to the Prolog script defined in step D (**50%**).
   1 You must complete this implementation individually, even if you are pair programming.
   2 Follow the same requirements defined in step C.